ROVPIA '99

PROCEEDINGS

INTERNATIONAL CONFERENCE
ON

ROBOTICS, VISION AND PARALLEL PROCESSING FOR AUTOMATION

JULY 16-18, 1999, IPOH, PERAK, MALAYSIA

KAMI MEMIMPIN

ISBN No. 983 - 9501 - 05 - 4

VOLUME 2

SCHOOL OF ELECTRICAL AND ELECTRONIC ENGINEERING
UNIVERSITY SAINS MALAYSIA
SRI ISKANDAR, 31750 TRONOH, PERAK, MALAYSIA

ROVPIA '99

KAMI MEMIMPIN

ROVPIA 1999

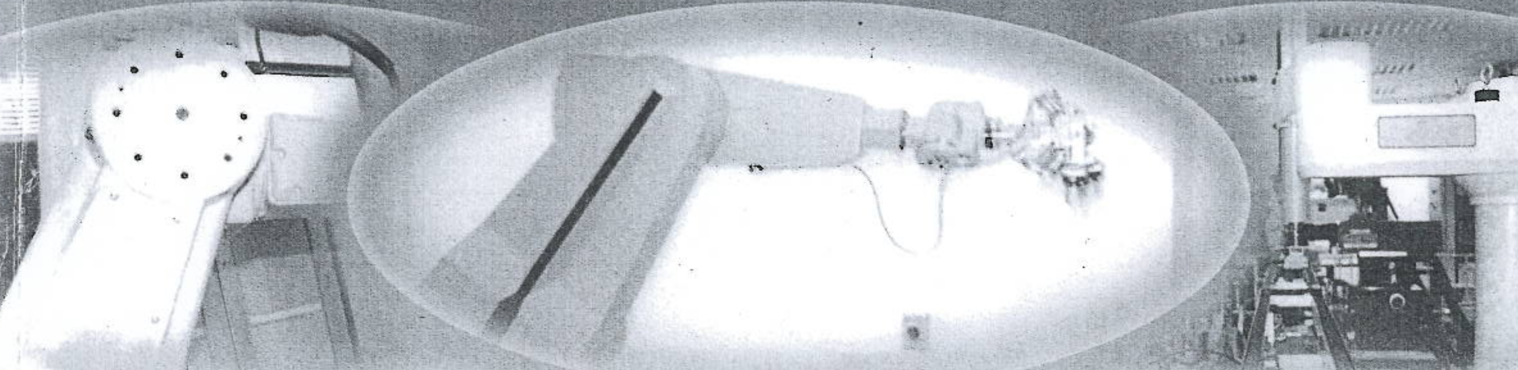
INTERNATIONAL CONFERENCE ON

ROBOTICS, VISION AND PARALLEL PROCESSING FOR AUTOMATION

16-18 July 1999, Ipoh, Malaysia

PROCEEDINGS
VOL 2

- Artificial Neural Networks
- Computer Vision
- Image Processing
- Parallel Processing
- Communications
- Robotics
- Control Systems
- Signal Processing
- Industrial Applications
- Microelectronics
- Power Systems



Organized by :
The School of Electrical and Electronic Engineering, Universiti Sains Malaysia, Perak Branch Campus, 31750 Tronoh, Perak Darul Ridzuan, Malaysia.

HYDRA:PARALLEL AND DISTRIBUTED Swarm COMPUTER ARCHITECTURE

Marcel O Odhiambo

Moi University
Department of Electrical and Communication Engineering
PO BOX 3900 Eldoret, Kenya
Tel. (254-321) 43292
Fax. (254-321) 43292
Email: dcse@net2000ke.com

Abstract

HYDRA: Parallel and Distributed Swarm computer architecture for agent processing is based on a single user Swarm computer architecture. Agents are self contained executable mobile processes that navigate a network of hosts using hosts' addresses, executing a task, and when the task terminates, the outcome of the task execution is sent to the user. HYDRA consists of four modules: the AP (Access Point) through which the user can access the HYDRA computer to run tasks, the RM (Routing Module), responsible for routing local agents, the PM (Processing Module), the execution engine in the HYDRA computer is the destination of an executing agent. Network Router (NR) responsible for routing agents between hosts on the network. The HYDRA computer architecture consists of several hosts connected together by an interconnection network to provide a distributed computer system where the PMs are distributed among several hosts on the network. Hosts on the network are grouped into domains, and the domains are interconnected together to form a network. Hosts in a domain are directly connected to every other host within that domain. Within a domain, a single host acts as a gateway host handling extra-domain agents. An interconnection protocol logically connects hosts in the same domain, builds a routing table at each host, and identifies a gateway host for each domain. A routing scheme using the routing table at each host ensures delivery of agents to their proper destinations, irrespective of the location of the Processing Module. Simulation tests on the the HYDRA computer prototype, and the preliminary test results obtained indicate that HYDRA computer architecture can be used to investigate the feasibility and characteristics of agents processing.

1 INTRODUCTION

HYDRA: Parallel and Distributed Swarm computer architecture is an extension of Swarm computer architecture [Errico 1996] to the distributed environment. Swarm computer was implemented and simulated on a single host. HYDRA computer builds hosts connection and message routing schemes in the Swarm computer so as to provide a distributed computer system for agents processing. HYDRA computer provides a distributed computer architecture that can be used to investigate the characteristics of agents execution:

1. Navigation - the capability to navigate the network using destination addresses.
2. Communication - the capability to carry data from one destination to another and thus, making it possible to communicate in the network.
3. The capability to access, and to modify stored data and thus, enable resource sharing in the network.
4. Parallel amd concurrent execution in the network.
5. Synchronisation when a shared resource is being accessed by multi-agents.

Agents are self contained executable mobile processes that navigate a network of hosts, using destination hosts' addresses. Agents cooperate in solving a task by sharing data relevant to the task. When an agent executes at a site, the agent store the shared data at the site, and any agent belonging to the same task can access and modify the stored data. When the task terminates, the agents collect the outcome of the task execution at each PM, and send the result to the user through console node or Access Point (AP). HYDRA computer architecture is constructed from a network of interconnected hosts which are grouped together into domains, using an interconnection protocol. A single host within each domain acts as the gateway host handling extra domain agents. A host in each domain is fully connected to every other host within that domain. Routing agents in the network requires a routing table containing entries of connections to every host in the same domain at each host. No host knows the network beyond its domain. Communication between the hosts is based on message-passing paradigm. Hosts in the network exchange messages[1] between them. The messages may be commands or agents to be executed by the destination host.

2 THE HYDRA COMPUTER

A number of agent systems have been discussed [Genesereth 1994, Jennings 1996, Reinhardt 1994, Wayner 1995], and a number of agent architectures have been implemented and tested [Errico 1996, White 1994, White 1996, Sapaty 1988, Sapaty 1991, Sapaty 1994]. The simulation test results obtained for the agent architectures point to the usefulness of agents in carrying tasks on the network. HYDRA computer architecture Figure 1, was implemented to explore agents execution in a distributed system. HYDRA computer is composed of four modules: Network Router (NR),Routing Module (RM), Processing Module (PM) and Access Point (AP) configured as individual processes that only communicate by passing short messages. The messages are a set of predefined commands that specify the action to be taken by the recipient host. Network Router interconnects hosts on the network, assigns hosts to the domains, identifies and marks gateway host for each domain, creates routing table for each host, and handles communications or agent routing to non-local destinations. Processing Module is the destination of an executing agent. PM receives and executes agents that are addressed to it. When an agent is to be sent to any destination, the PM packs the agent with the necessary information, i.e destination address etc, and sends it to RM for routing. Routing Module handles agents routing to local destinations within a PM. Agents with non-local destinations are passed to the Network Router. Access Point is the user interface to the system and also serves as the input/output of the HYDRA computer system. It is the entry point through which the user can access the system to run tasks or to issue commands to be executed. The user inputs commands or data to the HYDRA computer system through the AP, and the HYDRA computer system in turn uses the AP to display to the user the data generated as a consequence of a command or a task execution. HYDRA computer uses stream socket using TCP/IP transport protocols [Comer 1995].

Communication Router consists of the Network Router and the Routing Module. It connects and reliably delivers messages to all hosts in the network. At start up, the Communication Router connects and starts the routing module at each host, establishes a listening end-point and awaits message arrrival at the listening end points. The Network Router in a distributed network is required to reliably deliver messages to PMs both in remote and local hosts. Thus, the Network Router has to identify when communication is for the local or remote host. Communication is divided into local communication i.e routing agents to the local PM and remote communication i.e routing agents to the remote PMs.

[1]The term message or agent refers to the same entity and are used interchangeably throughout this document.

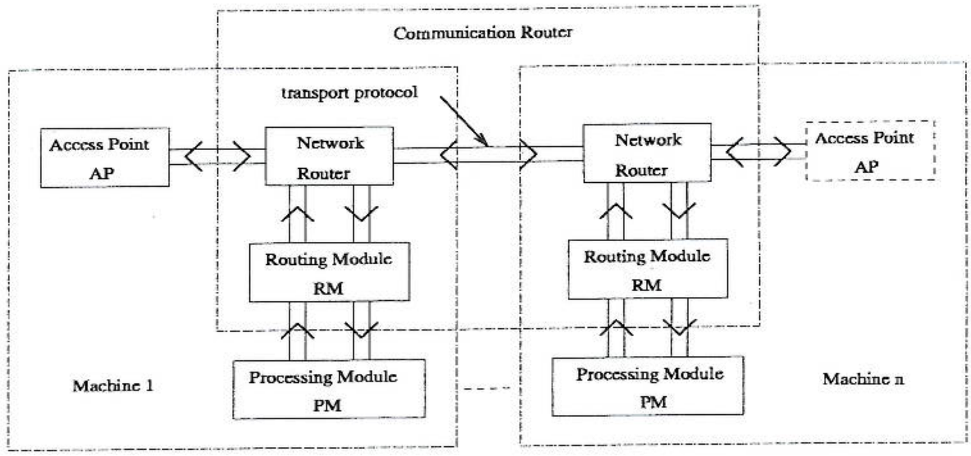


Figure 1: Prototype HYDRA Computer Architecture

Network Router (NR) connects host in the network and provides bi-directional communication channels for reliable, sequenced and ordered communication between hosts on the network.

Routing Module (RM) facilitates local routing of agents within the nodes in the Processing Module (PM).

Access Point (AP) enables the user to interface to the HYDRA system to execute commands or to run tasks. The user inputs data required by the tasks through the AP, likewise, the system displays the outcome/data generated as a consequence of command/task execution through the AP. This version of HYDRA computer architecture has been implemented for a single user thus, there is only one AP connected in the system.

Processing Module (PM) the execution engine carries the node structure or the distributed database where the agents are executed. In the agent paradigm, the execution environment is an active database represented by two concepts:

1. Processing Module and,
2. Node structure or Node space or active database.

2.1 Execution commands

No activity takes on the network until after the AP connects and issues commands to be executed. The commands from the following list:

CREATE_PM when the command followed by PM number is received by the RM, the referenced PM is created in accordance with the implementation protocol. The RM forks a child process using the UNIX 'fork' command, passes it the PM progran code and does the necessary house keeping.

DELETE_PM when the command followed by PM number is received, the RM searches its table for the PM and removes it. And, at the same time terminates the PM by sending it the UNIX 'kill' signal.

GET_NODES when the command followed by PM number is received, the RM searches its table for the PM, gets the node structure and sends to AP.

PUT_NODES when the command followed by PM number and the node structure is received, the RM searches its table for the PM, and loads or puts the node structure in the PM.

RUN_TASK when the command followed by program file is received, both RM and PM switch from command mode to execution mode and the PM begins to execute the program code.

CLOSE_AP or EXIT when the command is received, the Network Router disconnects any communication channels to AP and closes any open files. However, the Network Router remains executing in the background.

2.2 Interconnection Protocol

The interconnection protocol is a sequence of proceedures executed at each host when HYDRA computer system is started. The interconnection protocol connects hosts on the network. Three files are needed: config_file - contains domain hostnames, gateway_file - contains gateway hostnames and network_file - contains network hostnames. The interconnection protocol is best illustrated by a list of proceedures that are executed in sequence by the NR at each host.

1. Get the hostname (local_hostname) of the local host. The local_hostname is used by the local host to disqualify itself from identifying itself as a gateway host, and assigning itself a connection index. Read hostnames in both config_file and gateway_file, count the hosts in the two files. Though, a particular host may appear in both files, it is only counted once.

2. Intialise the structure host_array which contains various elements necessary for storing network host's information.

```
struct host_info {
  char *host_name;        /* hostname in config_file and gateway_file */
  int  host_addrtype;     /* host address type */
  int  host_addr_length; /* length of host address */
  char *host_addr;        /* host address */
  char *gateway_host;     /* marks host as gateway */
  int  connection;        /* connection index to host */
  int  host_level;        /* host level */
};
host_info *host_arr;
```

   The '*host_name' and 'connection' are used at each host to build a routing table for routing messages to remote hosts. After the structure has been filled every host in the network knows which are the gateway host(s), which host belongs to which host_level, network host's *host_addr, hostnames, host_addrtype and host_addr_length.

3. Create a listening socket by first creating a "socket" and then, calling "listen" function to await any event at the listening socket end-point.

4. Send connection requests to domain hosts or accept connection requests from domain hosts. After hosts have successfully sent and received connection request i.e hosts connectivity has been achieved, the element "connection" in the host_array structure is filled in. Each host has a unique 'connection' for every other host to which it is directly connected. 'connection' describes the connection or communication channel between the local host and the hosts within the domain.

5. Open and read hostnames from the network_file, and fill the elements in the structure host_data

```
struct machine_info {
    char *machine_names;
    int machine_num;
};
machine_info host_data;
```

   Hostnames read from the network_file are counted, assigned numbers (machine_num) 1, 2, 3,......n and *machine_names until all hosts have been assigned. The machine_num is unique for each host, thus, mapping unique numbers (PMs) to hosts in the network. If the read hostname from the network_file matches the local_hostname, the assigned machine_num is copied to local_machine_num. During execution in the network, each host checks if the message destination PM or address matches its local_machine_num, accepts the message and sends the message to the local PM. Otherwise, the destination address is a remote host, and the message is forwarded along the right channel towards the destination host.

6. The Network Router running at each host on the network forks a child process, and passes to it the Routing Module program code (separately compiled program code). The NR then goes into a loop continuously monitoring the listening end points for the user console (AP), the local PM and the network (connections from other hosts in the network) for any event or activity. At this stage the HYDRA computer has successfully been started (hosts on the network have been successfully interconnected, gateway hosts identified and marked, PMs mapped to hosts in the network, and the routing table built at each host). The HYDRA computer is set and waiting for the user to connect.

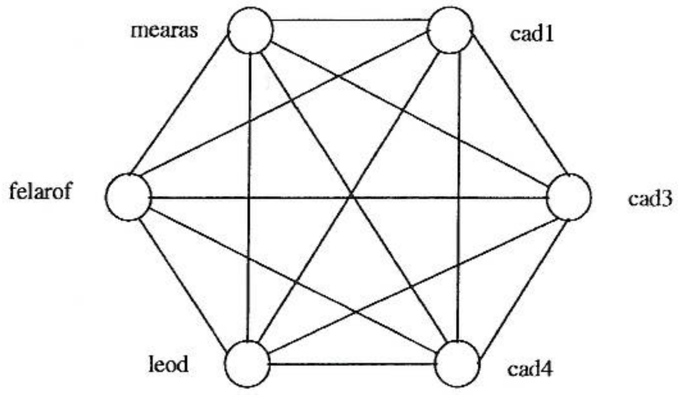


Figure 2: Network of six interconnected hosts in one domain

7. When the AP (console node or user console) connects to the network, the host to which the user is connected broadcasts its local_machine_num to all hosts (except itself) in the network. The other hosts recognise this local_machine_num as the address of the host to which the user messages should be sent.
8. The HYDRA system is now ready and set to receive user commands through the console node for execution.

2.3 Communication Strategy

Hosts on the network are only aware of their immediate neighbours to which they are connected, but knows nothing about the rest of the network. This has been achieved by grouping hosts into domains. Nevertheless, every domain has a gateway host that handles extra domain messages. The communication strategy adopted is that:

"If a message is for the local host or a host within the sending host's domain then, send the message to its destination. Otherwise, route the message to the gateway host".

The strategy is similar to *"group communication in Amoeba"* [Tanenbaum 1995, Coulouris 1994]

Message routing between host in the network requires a search table, a routing table, and a message routing scheme. The message routing scheme maps a destination PM to a host on the network. The search table translates the PM number to a hostname. The search table (host_data in section 2.2 contains hostnames (*machine_names) and the assigned PM numbers (machine_num).

3 SIMULATION TESTS AND RESULTS

Simulation tests were run to test two aspects of HYDRA computer architecture. The first test investigated the ability to connect hosts on the network and to route agents between hosts. The second test investigated the feasibility of using the HYDRA computer architecture and the agent technology to do useful work, and to investigate agents characteristics listed in the introductoion. The first test was used to connect hosts in Figure 2 and Figure 3. The simulation test results for connection and message routing in network of Figure 3(b) are shown in Figure 4. The second test was performed on network Figure 5.

The routine loads a value to agent workspace variable AW2 (a variable carried by the agent), inserts a destination address (node 1 in all PMs in the network) in the destination list (DL) and then, sends the agent to the address in the destination list. At the destination address, the agent loads a value from the persistent workspace PW3, adds this value to the value carried in the agent variable and stores the result in the AW2, the loop is repeated five times. After exiting the loop, the agent stores the result in PW3 and AW2. The agent inserts destination address (PM 3 node 1) in the destination list, and navigates to this address. On reaching the address in the DL, the agent loads the value from persistent workspace PW3, add this value to the value in the AW2, and stores the result in AW3. The agent then, loads the value in persistent workspace PW4, adds it to the value in the AW3, and store the final result in persistent workspace PW4. The agent then, terminates.

3.1 Task Execution Test Results

The test result are shown in Figure 6. At the end of the addition loop, the agent stored the final result in persistent workspace PW3 and agent workspace AW2. The agent then, inserted the

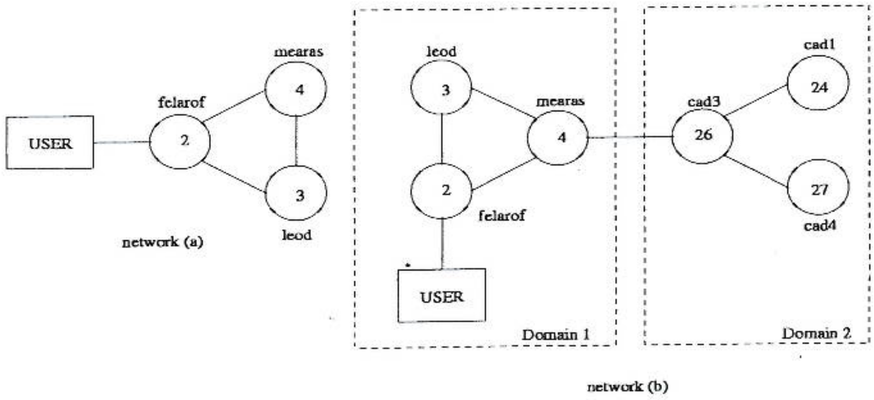


Figure 3: Network of six interconnected hosts in two domains

router.log

Network Router started on host number 26	Routing Module started on host number 2
Sun May 25 21:54:54 1997	Sun May 25 22:03:56 1997
Network Router started on host number 24	Routing Module started on host number 3
Sun May 25 21:54:55 1997	Sun May 25 22:03:57 1997
Network Router started on host number 27	Routing Module started on host number 4
Sun May 25 21:54:55 1997	Sun May 25 22:04:32 1997
Network Router started on host number 3	Successful data transfer to PM: 2
Sun May 25 21:54:55 1997	Sun May 25 22:09:19 1997
Network Router started on host number 2	Successful data transfer to PM: 3
Sun May 25 21:54:56 1997	Sun May 25 22:09:27 1997
Network Router started on host number 4	Successful data transfer to PM: 4
Sun May 25 21:59:26 1997	Sun May 25 22:09:38 1997
Routing Module started on host number 26	Successful data transfer to PM: 24
Sun May 25 21:54:55 1997	Sun May 25 22:10:34 1997
Routing Module started on host number 24	Successful data transfer to PM: 26
Sun May 25 21:54:56 1997	Sun May 25 22:10:49 1997
Routing Module started on host number 27	Successful data transfer to PM: 27
Sun May 25 21:54:55 1997	Sun May 25 22:10:59 1997

Figure 4: Hosts interconnection and message routing in Figure 3(b)

destination address (PM3 node 1) in the destination list and navigated to this address. On reaching its destination in Figure 6(b), the agent executed the following loops:

In the 1st loop, the agent from PM1 loads the contents of PW3 into AW3, adds the contents of AW2 to AW3 and stores the sum in AW3. The agent then, loads the contents of PW4 into AW4, adds the contents of AW3 to the contents of AW4 and stores the sum in PW4.
In the 2nd loop, the agent from PM2 loads the contents of PW3 into AW3, adds the contents of AW2 to AW3 and stores the sum in AW3. The agent then, loads the contents of PW4 into AW4, adds the contents of AW3 to the contents of AW4 and stores the sum in PW4.
In the 3rd loop, the agent from PM3 loads the contents of PW3 into AW3, adds the contents of AW2 to AW3 and stores the sum in AW3. The agent then, loads the contents of PW4 into AW4, adds the contents of AW3 to the contents of AW4 and stores the sum in PW4.

```
---------------------------------------------------------------------
Executing the task manually for test1 at the PMs.

PM1            PM2            PM3

22             23             24      value in PW3
x5             x5             x5
----           ----           ----
AA             AF             B4      value in PW3 multiplied five times
```

```
+1A             +1A              +1A      value in AW2
----            ----             ----
 C4              C9               CE      final value stored in PW3

All agents propogate to PM3 node1.

Value from     PM1  C4     Value from     PM2  C9     Value from     PM3  CE
Value from     PM3  CE     Value from     PM3  CE     Value from     PM3  CE
                    ----                       ----                       ----
sub-total          192                        197                        19C

Total            192  +  197 +  19C  =  4C5
```

3.2 Agents Characteristics

Section 1.0 lists the characteristics that agents executing in a distributed system should exhibit. I shall refer to test results Figure 6.

1. Navigation - a means by which agents traverse the network using destination addresses. From the console node the agent was sent to address $xFFFF0001_{hex}$[2]. The address FFFF in the PM field signifies a broadcast address (all PMs in the network). From the test results in Figure 6, it can be observed that the agents navigated the network using destination addresses, reached their destinations and executed the task.
2. Communication - the capability to carry data from one destination to another in the network. The agent carried the parameter $1A_{hex}$ in AW2 from the console node to node 1 PMs1–3 in the network, thus, making it possible to communicate in the network. Therefore, proving that agents can be used to pass or communicate messages in a distributed system.
3. The capability to access and to modify a stored value at a given site. The agent accessed the value stored at persistent workspace PW3 in node 1 in PMs1–3 on the network. The unique value at the respective sites, made it possible to observe the values retrieved by the agent at each site being used in the addition loop. The result show that the HYDRA computer architecture provides agents with the means by which a shared resource can be accessed by agents executing in the system.
4. Concurrent execution by the agents. From the console node the agents were concurrently sent to node1 in PMs1–3, on reaching their destination, the agents independently executed the addition loop Figure 6(a), and the results stored at the local site. This proves that HYDRA computer architecture supports parallel and concurrent execution.
5. Synchronisation - when a shared resource is being accessed and modified by several agents. In Figure 6(b), the agents access and modify a value stored at PM3. The sum total of 000004C5 is the sum total of the results in PW3 in PM1 and PM2 individually added to the value in PW3 in PM3 node1. Without synchronisation when executing agents access a shared resource, the final figures would not be the same as the sum obtained by manual calculation.

[2] a destination address consists of PM address and node address

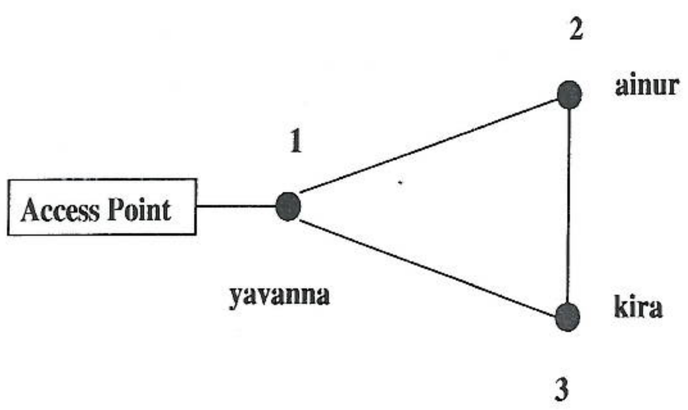


Figure 5: Network for task execution test

	PM1 logfile	PM2 logfile	PM3 logfile	loop_count
value in persistent workspace PW3	22	23	24	
value in AW2	0000001A	0000001A	0000001A	
Total	0000003C	0000003D	0000003E	1
Total	0000005E	00000060	00000062	2
Total	00000080	00000083	00000086	3
Total	000000A2	000000A6	000000AA	4
Total	000000C4	000000C9	000000CE	5

(a) The agent adds the contents of AW2 to the contents of PW3, the agent goes executes the addition loop 5 times

loop_count	PM1	PM2	PM3	Sub Total	Total
1	000000C4 (AW2)	—	000000CE (PW3)	00000192 (AW3)	00000192 (PW4)
2	—	000000C9 (AW2)	000000CE (PW3)	00000197 (AW3)	00000329 (PW4)
3	—	—	000000CE (PW3) 000000CE (AW2)	0000019C (AW3)	000004C5 (PW4)

(b) The results obtained from every PM in the network are added up and stored at PW4 in PM3 node 1

Figure 6: Task execution results - test program 1

4 CONCLUSION

HYDRA computer aimed to implement a distributed computer system based on Swarm computer architecture on which agents processing can be explored. The simulation tests covered hosts interconnections, message routing and task execution in the HYDRA computer system. The test results obtained show that the interconnection and message schemes implemented were able to connect host on the network and to route agents between the host irrespective of the location of the destination address, whether local or remote. The task execution test showed that agents can used do some useful work, and HYDRA prototype provides a distributed computer architecture that supports distributed agents processing.

References

[Comer 1995] COMER DOUGLAS, E. *Internetworking with TCP/IP, Vol. 1, Principles, 3rd edition Protocols and Architecture.* Prentice-Hall, Inc. Englewood Cliffs, New Jersey 07632, 1995.

[Coulouris 1994] COULOURIS, G. .F., DOLLIMORE, J., KINDBERG, K. *Distributed Systems: Concepts and Design.* Addison-Wesley Publishing Company, 1994.

[Errico 1996] LUCIANO de ERRICO Agent Based Distributed Parallel Processing. *A Dissertation for the degree of Doctorate of Philosophy, Department of Electrical and Electronic Engineering, University of Surrey. January 17, 1996.*

[Genesereth 1994] GENESERETH, MICHAEL R., and KETCHPE, STEVEN P. Software Agents. *Comms. of the ACM, July 1994, Vol.37, No.7, pp. 48—54.* 1994.

[Jennings 1996] JENNINGS, NICK and WOOLDRIDGE, MICHAEL. Software Agents. *IEE Review, January 1996, pp. 17—20*

[Reinhardt 1994] REINHARDTT, ANDY. The Networks with Smarts. *BYTE October 1994, pp. 51—64.*

[Sapaty 1988] SAPATY, P. S. WAVE-1: A New Ideology of Parallel and Distributed Processing on Graphs and Newtorks. *Technical report, V. M Glushkov Institute of Cybernetics, Ukrainian Academy of Sciences, Kiev USSR 1988.*

[Sapaty 1991] SAPATY, P. S. Logic Flow in Active Data. *In VLSI for Artificial Intelligence and Neural Networks, edited by Delgado-Frias J. G. and Moore W. R, Plenum Press, New York, 1991.*

[Sapaty 1994] SAPATY, P. S., BORST P. M. An Overview of WAVE Language and Systems for Distributed Processing in Open Networks. *University of Surrey, Department of Electronic and Electrical Engineering, Technical report, June 1994.*

[Tanenbaum 1995] ANDREW S TANENBAUM *Distributed Operating Systems.* Prentice-Hall International Inc. 1995.

[Wayner 1995] WAYNER, PETER. *AGENTS UNLEASHED A Public Domain Look at Agent Technology.* AP PROFESSIONAL 1995.

[White 1994] WHITE, JAMES E. Telescript Technology: The Foundation for the Electronic Marketplace. *General Magic white paper.* General Magic Inc. 2465 Latham Street, Mountain View, California 94040.

[White 1996] WHITE, JAMES E. Telescript Technology: Mobile Agents. *General Magic white paper.* General Magic Inc. 2465 Latham Street, Mountain View, California 94040.